Logit, Probit, and Multinomial Logit models in R

(v. 3.5)

Oscar Torres-Reyna

otorres@princeton.edu

December 2014

http://www.princeton.edu/~otorres/

If outcome or dependent variable is binary and in the form 0/1, then use logit or probit models. Some examples are:	
Did you vote in the last election? 0 'No' 1 'Yes'	Do you prefer to use public transportation or to drive a car? 0 'Prefer to drive' 1 'Prefer public transport'

If outcome or dependent variable is categorical but are ordered (i.e. low to high), then use ordered logit or ordered probit models. Some examples are:	
Do you agree or disagree with the President? 1 'Disagree' 2 'Neutral' 3 'Agree'	What is your socioeconomic status? 1 'Low' 2 'Middle' 3 'High'

If outcome or dependent variable is categorical without any particular order, then use multinomial logit. Some examples are:	
If elections were held today, for which party would you vote? 1 'Democrats' 2 'Independent' 3 'Republicans'	What do you like to do on the weekends? 1 'Rest' 2 'Go to movies' 3 'Exercise'

Logit model

```
# Getting sample data
library(foreign)
mydata <- read.dta("https://dss.princeton.edu/training/Panel101.dta")

# Running a logit model
logit <- glm(y_bin ~ x1 + x2 + x3, family=binomial(link="logit"), data=mydata)
```

Store results | **Outcome** | **Predictors** | **Type of model** | **Data source**

```
summary(logit)
```

```
Call:
glm(formula = y_bin ~ x1 + x2 + x3, family = binomial(link = "logit"),
    data = mydata)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.0277   0.2347   0.5542   0.7016   1.0839

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   0.4262     0.6390   0.667   0.5048
x1            0.8618     0.7840   1.099   0.2717
x2            0.3665     0.3082   1.189   0.2343
x3            0.7512     0.4548   1.652   0.0986 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 70.056  on 69  degrees of freedom
Residual deviance: 65.512  on 66  degrees of freedom
AIC: 73.512

Number of Fisher Scoring iterations: 5
```

The **Pr(>|z|)** column shows the two-tailed p-values testing the null hypothesis that the coefficient is equal to zero (i.e. no significant effect). The usual value is 0.05, by this measure none of the coefficients have a significant effect on the log-odds ratio of the dependent variable. The coefficient for `x3` is significant at 10% (<0.10).

The **z value** also tests the null that the coefficient is equal to zero. For a 5% significance, the z-value should fall outside the ±1.96.

The **Estimate** column shows the coefficients in log-odds form. When `x3` increase by one unit, the expected change in the log odds is 0.7512. What you get from this column is whether the effect of the predictors is positive or negative. See next page for an extended explanation.

Logit model

```
# The stargazer() function from the package –stargazer allows a publication quality of the logit model.
# The model will be saved in the working directory under the name 'logit.htm' which you can open with Word or any other word processor.

library(stargazer)
stargazer(logit, type="html", out="logit.htm")
```

	Dependent variable:
	y_bin
x1	0.862
	(0.784)
x2	0.367
	(0.308)
x3	0.751*
	(0.455)
Constant	0.426
	(0.639)
Observations	70
Log Likelihood	-32.756
Akaike Inf. Crit.	73.512
Note:	*p**p***p<0.01

NOTE: Use the option `type = "text"` if you want to see the results directly in the RStudio console.

Logit model: odds ratio

Odds ratio interpretation (OR): Based on the output below, when x3 increases by one unit, the odds of y = 1 increase by 112% -(2.12-1)*100-. Or, the odds of y =1 are 2.12 times higher when x3 increases by one unit (keeping all other predictors constant). To get the odds ratio, you need explonentiate the logit coefficient.

```
# Estimating the odds ratio by hand

cbind(Estimate=round(coef(logit),4),
      OR=round(exp(coef(logit)),4))

             Estimate     OR
(Intercept)   0.4262 1.5314
x1            0.8618 2.3674
x2            0.3665 1.4427
x3            0.7512 2.1196
```

```
# Using package --mfx--

library(mfx)
logitor(y_bin ~ x1 + x2 + x3, data=mydata)
Call:
logitor(formula = y_bin ~ x1 + x2 + x3, data = mydata)

Odds Ratio:
   OddsRatio Std. Err.      z   P>|z|
x1   2.36735   1.85600 1.0992 0.27168
x2   1.44273   0.44459 1.1894 0.23427
x3   2.11957   0.96405 1.6516 0.09861 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The **Estimate** column shows the coefficients in log-odds form. When x3 increase by one unit, the expected change in the log odds is 0.7512. Lets hold x1 and x2 constant at their means, and vary x3 with values 1, 2, and 3, to get the predicted log-odds given each of the three values of x3:

```
r1 <- logit$coeff[1] + logit$coeff[2]*mean(mydata$x1) +
                       logit$coeff[3]*mean(mydata$x2) +
                       logit$coeff[4]*1
> r1
   1.784902

r2 <- logit$coeff[1] + logit$coeff[2]*mean(mydata$x1) +
                       logit$coeff[3]*mean(mydata$x2) +
                       logit$coeff[4]*2
> r2
   2.536113

r3 <- logit$coeff[1] + logit$coeff[2]*mean(mydata$x1) +
                       logit$coeff[3]*mean(mydata$x2) +
                       logit$coeff[4]*3
> r3
   3.287325
```

When x3 increases from 1 to 2, the log-odds increases:

```
r2-r1
    0.7512115
```

When x3 increases from 2 to 3, the log-odds increases:

```
r3-r2
    0.7512115
```

Which corresponds to the estimate for x3 above.

The odds ratio, is the exponentiation of the difference of the log-odds

```
> exp(r2-r1)
    2.119566
```

Or, the ratio of the exponentiation of each of the log-odds.

```
> exp(r2)/exp(r1)
    2.119566
```

Which corresponds to the OR value for x3 above.

Logit model: odds ratios

```
# Relative risk ratios allow an easier interpretation of the logit coefficients. They are the
exponentiated value of the logit coefficients.

logit.or = exp(coef(logit))
logit.or
(Intercept)          x1          x2          x3
   1.531417    2.367352    1.442727    2.119566

library(stargazer)
stargazer(logit, type="html", coef=list(logit.or), p.auto=FALSE, out="logitor.htm")
```

	Dependent variable:
	y_bin
x1	2.367
	(0.784)
x2	1.443
	(0.308)
x3	2.120*
	(0.455)
Constant	1.531
	(0.639)
Observations	70
Log Likelihood	-32.756
Akaike Inf. Crit.	73.512
Note:	*p**p***p<0.01

Keeping all other variables constant, when x1 increases one unit, it is 2.367 times more likely to be in the 1 category. In other words, the odds of being in the 1 category (as opposed to the 0 category) are 136% higher when x1 move one unit (2.36 – 1). The coefficient, however, is not significant.

NOTE: Use the option `type = "text"` if you want to see the results directly in the RStudio console.

Logit model: predicted probabilities

The logit model can be written as (Gelman and Hill, 2007):

$$\Pr(y_i = 1) = \mathrm{Logit}^{-1}(X_i\beta)$$

In the example:

```
logit <- glm(y_bin ~ x1 + x2 + x3, family=binomial(link="logit"), data=mydata)

coef(logit)
         (Intercept)          x1          x2          x3
          0.4261935   0.8617722   0.3665348   0.7512115
```

$$\Pr(y_i = 1) = \mathrm{Logit}^{-1}(0.4261935 + 0.8617722*x1 + 0.3665348*x2 + 0.7512115*x3)$$

Estimating the probability at the mean point of each predictor can be done by inverting the logit model. Gelman and Hill provide a function for this (p. 81), also available in the R package `-arm-`

```
invlogit = function (x) {1/(1+exp(-x))}
 invlogit(coef(logit)[1]+
           coef(logit)[2]*mean(mydata$x1)+
           coef(logit)[3]*mean(mydata$x2)+
           coef(logit)[4]*mean(mydata$x3))
```

$$\Pr(y_i = 1) = 0.8328555$$

Logit model: predicted probabilities

Adding categorical variable, the model would be:

```
logit.cat <- glm(y_bin ~ x1 + x2 + x3 + opinion,
                              family=binomial(link="logit"), data=mydata)
coef(logit.cat)
  (Intercept)           x1           x2           x3   opinionAgree   opinionDisag opinionStr disag
    0.8816118    1.1335562    0.3021217    0.3976276     -1.9163569      0.3270627        0.6891686
```

Estimating the probability when opinion = 'Agree'

```
invlogit = function (x) {1/(1+exp(-x))}

invlogit(coef(logit.cat)[1]+
           coef(logit.cat)[2]*mean(mydata$x1)+
           coef(logit.cat)[3]*mean(mydata$x2)+
           coef(logit.cat)[4]*mean(mydata$x3)+
           coef(logit.cat)[5]*1)
```

$Pr(y_i = 1 |$ opinion= "Agree"$) = 0.5107928$

Logit model: predicted probabilities

```
invlogit = function (x) {1/(1+exp(-x))}
```

Estimating the probability when opinion = '**Disagree**'

```
invlogit(coef(logit.cat)[1]+
         coef(logit.cat)[2]*mean(mydata$x1)+
         coef(logit.cat)[3]*mean(mydata$x2)+
         coef(logit.cat)[4]*mean(mydata$x3)+
         coef(logit.cat)[6]*1)
```

$\Pr(y_i = 1 |$ opinion= "Disagree") = 0.9077609

Estimating the probability when opinion = '**Strongly disagree**'

```
invlogit(coef(logit.cat)[1]+
         coef(logit.cat)[2]*mean(mydata$x1)+
         coef(logit.cat)[3]*mean(mydata$x2)+
         coef(logit.cat)[4]*mean(mydata$x3)+
         coef(logit.cat)[7]*1)
```

$\Pr(y_i = 1 |$ opinion= "Strongly disagree") = 0.933931

Estimating the probability when opinion = '**Strongly agree**'

```
invlogit(coef(logit.cat)[1]+
         coef(logit.cat)[2]*mean(mydata$x1)+
         coef(logit.cat)[3]*mean(mydata$x2)+
         coef(logit.cat)[4]*mean(mydata$x3))
```

$\Pr(y_i = 1 |$ opinion= "Strongly agree") = 0.8764826

Logit model: predicted probabilities

Another way to estimate the predicted probabilities is by setting initial conditions.

Getting predicted probabilities holding all predictors or independent variables to their means.

```
allmean <-  data.frame(x1=mean(mydata$x1),
                       x2=mean(mydata$x2),
                       x3=mean(mydata$x3))
allmean
         x1        x2       x3
1 0.6480006 0.1338694 0.761851
```

Creating a new dataset with the mean values of the predictors

After estimating the logit model and creating the dataset with the mean values of the predictors, you can use the `predict()` function to estimate the predicted probabilities (for help/details type `?predict.glm`), and add them to the `allmean` dataset.

```
allmean$pred.prob <- predict(logit, newdata=allmean, type="response")
```

The object with the logit coefficients | **Dataset with the conditions** | **Requesting predicted probabilities**

```
allmean
         x1        x2       x3 pred.prob
1 0.6480006 0.1338694 0.761851 0.8328555
```

When all predictor values are hold to their means, the probability of y = 1 is 83%.

Logit model: predicted probabilities with categorical variable

```
logit <- glm(y_bin ~ x1+x2+x3+opinion, family=binomial(link="logit"), data=mydata)
```

To estimate the predicted probabilities, we need to set the initial conditions. Getting predicted probabilities holding all predictors or independent variables to their means for each category of categorical variable ‘opinion’:

```
allmean <-  data.frame(x1=rep(mean(mydata$x1),4),
                       x2=rep(mean(mydata$x2),4),
                       x3=rep(mean(mydata$x3),4),
                       opinion=as.factor(c("Str agree","Agree","Disag","Str disag")))
allmean
```

Creating a new dataset with the mean values of the predictors for each category

```
 x1         x2        x3    opinion
1 0.6480006 0.1338694 0.761851 Str agree
2 0.6480006 0.1338694 0.761851     Agree
3 0.6480006 0.1338694 0.761851     Disag
4 0.6480006 0.1338694 0.761851 Str disag
```

```
allmean <- cbind(allmean,predict(logit, newdata=allmean, type="response", se.fit=TRUE))
```

The object with the logit coefficients | **Dataset with the conditions** | **Requesting predicted probabilities** | **Standard error of the prediction**

```
allmean
         x1        x2       x3   opinion       fit     se.fit residual.scale
1 0.6480006 0.1338694 0.761851 Str agree 0.8764826 0.07394431              1
2 0.6480006 0.1338694 0.761851     Agree 0.5107928 0.15099064              1
3 0.6480006 0.1338694 0.761851     Disag 0.9077609 0.06734568              1
4 0.6480006 0.1338694 0.761851 Str disag 0.9339310 0.06446677              1
```

(continue next page)

Logit model: predicted probabilities with categorical variable

```
# Renaming "fit" and "se.fit" columns
names(allmean)[names(allmean)=="fit"] = "prob"

names(allmean)[names(allmean)=="se.fit"] = "se.prob"

# Estimating confidence intervals

allmean$ll = allmean$prob - 1.96*allmean$se.prob

allmean$ul = allmean$prob + 1.96*allmean$se.prob

allmean

         x1        x2       x3   opinion      prob    se.prob residual.scale        ll        ul
1 0.6480006 0.1338694 0.761851 Str agree 0.8764826 0.07394431              1 0.7315518 1.0214134
2 0.6480006 0.1338694 0.761851     Agree 0.5107928 0.15099064              1 0.2148511 0.8067344
3 0.6480006 0.1338694 0.761851     Disag 0.9077609 0.06734568              1 0.7757634 1.0397585
4 0.6480006 0.1338694 0.761851 Str disag 0.9339310 0.06446677              1 0.8075762 1.0602859
```

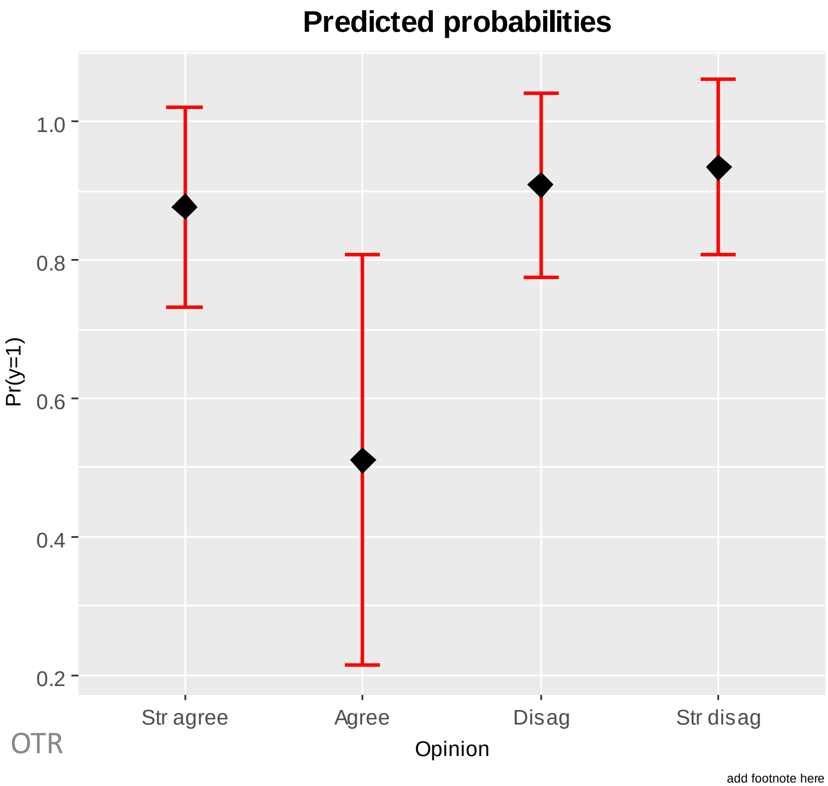
(continue next page)

Logit model: predicted probabilities with categorical variable

```
# Plotting predicted probabilities and confidence intervals using ggplot2

library(ggplot2)

ggplot(allmean, aes(x=opinion, y = prob)) +
  geom_errorbar(aes(ymin = ll, ymax = ul), width = 0.2, lty=1, lwd=1, col="red") +
  geom_point(shape=18, size=5, fill="black") +
  scale_x_discrete(limits = c("Str agree","Agree","Disag","Str disag")) +
  labs(title= " Predicted probabilities", x="Opinion", y="Pr(y=1)", caption = "add footnote here") +
  theme(plot.title = element_text(family = "sans", face="bold", size=13, hjust=0.5),
        axis.title = element_text(family = "sans", size=9),
        plot.caption = element_text(family = "sans", size=5))
```



Logit model: marginal effects

```
# Using package -mfx-
# See http://cran.r-project.org/web/packages/mfx/mfx.pdf
install.packages("mfx")  #Do this only once
library(mfx)
logitmfx(y_bin ~ x1+x2+x3, data=mydata)
Call:
logitmfx(formula = y_bin ~ x1 + x2 + x3, data = mydata)

Marginal Effects:
      dF/dx Std. Err.      z   P>|z|
x1 0.119965  0.104836 1.1443 0.25249
x2 0.051024  0.041155 1.2398 0.21504
x3 0.104574  0.053890 1.9405 0.05232 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Marginal effects show the change in probability when the predictor or independent variable increases by one unit. For continuous variables this represents the instantaneous change given that the 'unit' may be very small. For binary variables, the change is from 0 to 1, so one 'unit' as it is usually thought.

Ordinal logit model

```
# Getting sample data
library(foreign)
mydata <- read.dta("https://dss.princeton.edu/training/Panel101.dta")

# Loading library -MASS-
library(MASS)

# Running the ordered logit model
m1 <- polr(opinion ~ x1 + x2 + x3, data=mydata, Hess=TRUE)
```

Store results | Outcome | Predictors | Data source | Required for SE

```
summary(m1)
Call:
polr(formula = opinion ~ x1 + x2 + x3, data = mydata, Hess = TRUE)

Coefficients:
     Value Std. Error t value
x1 0.98140     0.5641  1.7397
x2 0.24936     0.2086  1.1954
x3 0.09089     0.1549  0.5867

Intercepts:
                Value   Std. Error t value
Str agree|Agree -0.2054  0.4682    -0.4388
Agree|Disag      0.7370  0.4697     1.5690
Disag|Str disag  1.9951  0.5204     3.8335

Residual Deviance: 189.6382
AIC: 201.6382
```

Ordinal logit model: p-values

```
# Getting coefficients and p-values

m1.coef <- data.frame(coef(summary(m1)))

m1.coef$pval = round((pnorm(abs(m1.coef$t.value), lower.tail = FALSE) * 2),2)

m1.coef

                        Value Std..Error    t.value pval
x1                 0.98139603  0.5641136  1.7397134 0.08
x2                 0.24935530  0.2086027  1.1953599 0.23
x3                 0.09089175  0.1549254  0.5866807 0.56
Str agree|Agree -0.20542664  0.4682027 -0.4387558 0.66
Agree|Disag        0.73696754  0.4696907  1.5690486 0.12
Disag|Str disag  1.99507902  0.5204282  3.8335334 0.00
```

Ordered logit model

```
# The stargazer() function from the package –stargazer allows a publication quality of the logit model.
# The model will be saved in the working directory under the name 'm1.htm' which you can open with Word or any other word processor.

library(stargazer)
stargazer(m1, type="html", out="m1.htm")
```

	Dependent variable:
	opinion
x1	0.981*
	(0.564)
x2	0.249
	(0.209)
x3	0.091
	(0.155)
Observations	70
Note:	*p**p***p<0.01

NOTE: Use the option `type = "text"` if you want to see the results directly in the RStudio console.

Ordered logit model: odds ratios

```
# Relative risk ratios allow an easier interpretation of the logit coefficients. They are the
exponentiated value of the logit coefficients.

m1.or=exp(coef(m1))
m1.or
      x1       x2       x3
2.668179 1.283198 1.095150

library(stargazer)
stargazer(m1, type="html", coef=list(m1.or),  p.auto=FALSE, out="m1or.htm")
```

	Dependent variable:
	opinion
x1	$\boxed{2.668^{*}}$
	(0.564)
x2	1.283
	(0.209)
x3	1.095
	(0.155)
Observations	70
Note:	*p**p***p<0.01

Keeping all other variables constant, when x1 increases one unit, it is 2.668 times more likely to be in a higher category. In other words, the odds of moving to a higher category in the outcome variable is 166% when x1 move one unit (2.66 – 1). The coefficient is significant.

NOTE: Use the option `type = "text"` if you want to see the results directly in the RStudio console.

Ordinal logit model: predicted probabilities

```
# Use "probs" for predicted probabilities

m1.pred <- predict(m1, type="probs")
summary(m1.pred)

 Str agree          Agree             Disag            Str disag
 Min.   :0.1040   Min.   :0.1255   Min.   :0.1458   Min.   :0.07418
 1st Qu.:0.2307   1st Qu.:0.2038   1st Qu.:0.2511   1st Qu.:0.17350
 Median :0.2628   Median :0.2144   Median :0.2851   Median :0.23705
 Mean   :0.2869   Mean   :0.2124   Mean   :0.2715   Mean   :0.22923
 3rd Qu.:0.3458   3rd Qu.:0.2271   3rd Qu.:0.2949   3rd Qu.:0.26968
 Max.   :0.5802   Max.   :0.2313   Max.   :0.3045   Max.   :0.48832
```

The bold numbers are the predicted probabilities of each category when all predictors are at their mean value

Ordinal logit model: predicted probabilities

```
# At specific values, example x1 and x2 at their means, and x3 = 1 and x3 = 2.
# Use "probs" for predicted probabilities given specific predictors
setup1 <-  data.frame(x1=rep(mean(mydata$x1),2),
                      x2=rep(mean(mydata$x2),2),
                      x3=c(1,2))
setup1
         x1        x2 x3
1 0.6480006 0.1338694  1
2 0.6480006 0.1338694  2
```

Setup for new predicted probabilities

```
setup1[, c("pred.prob")] <- predict(m1, newdata=setup1, type="probs")
setup1
         x1        x2 x3 pred.prob.Str agree pred.prob.Agree pred.prob.Disag pred.prob.Str disag
1 0.6480006 0.1338694  1           0.2757495       0.2184382       0.2804806           0.2253318
2 0.6480006 0.1338694  2           0.2579719       0.2135235       0.2869123           0.2415923

# Use "class" for the predicted category
setup1[, c("pred.prob")] <- predict(m1, newdata=setup1, type="class")
setup1
         x1        x2 x3 pred.prob
1 0.6480006 0.1338694  1     Disag
2 0.6480006 0.1338694  2     Disag
```

These are the predicted categories given the new data

Ordinal logit model: marginal effects

```
# Load package "erer", use function ocMe() for marginal effects

library(erer)

x <- ocME(m1, x.mean=TRUE)

x

   effect.Str agree effect.Agree effect.Disag effect.Str disag
x1           -0.198       -0.047        0.076            0.169
x2           -0.050       -0.012        0.019            0.043
x3           -0.018       -0.004        0.007            0.016

# Type the following if you want t and p-values

x$out
```

Multinomial logit model

```
# Loading the required packages

library(foreign)
library(nnet)
library(stargazer)

# Getting the sample data from UCLA

mydata = read.dta("http://www.ats.ucla.edu/stat/data/hsb2.dta")

# Checking the output (dependent) variable

table(mydata$ses)

   low middle   high
   47     95     58

# By default the first category is the reference.
# To change it so 'middle' is the reference type

mydata$ses2 = relevel(mydata$ses, ref = "middle")
```

NOTE: This section is based on the UCLA website http://www.ats.ucla.edu/stat/r/dae/mlogit.htm, applied to data from the page http://www.ats.ucla.edu/stat/stata/output/stata_mlogit_output.htm. Results here reproduce the output in the latter to compare, and to provide an additional source to interpret outcomes.

Multinomial logit model

```
# Running the multinomial logit model using the multinom() function

multi1 = multinom(ses2 ~ science + socst +  female, data=mydata)
```

Store results | Outcome | Predictors | Data source

```
summary(multi1)

Call:
multinom(formula = ses2 ~ science + socst + female, data = mydata)

Coefficients:
     (Intercept)     science       socst  femalefemale
low     1.912288 -0.02356494 -0.03892428    0.81659717
high   -4.057284  0.02292179  0.04300323   -0.03287211

Std. Errors:
     (Intercept)    science      socst  femalefemale
low     1.127255 0.02097468 0.01951649    0.3909804
high    1.222937 0.02087182 0.01988933    0.3500151

Residual Deviance: 388.0697
AIC: 404.0697
```

These are the logit coefficients relative to the reference category. For example, under 'science', the -0.02 suggests that for one unit increase in 'science' score, the logit coefficient for 'low' relative to 'middle' will go down by that amount, -0.02.
In other words, if your science score increases one unit, your chances of staying in the middle ses category are higher compared to staying in low ses.

Multinomial logit model

 # The multinom() function does not provide p-values, you can get significance of the coefficients using the stargazer() function from the package –stargazer.

 # The model will be saved in the working directory under the name 'multi1.htm' which you can open with Word or any other word processor.

```
library(stargazer)
stargazer(multi1, type="html", out="multi1.htm")
```

	Dependent variable:	
	low	high
	(1)	(2)
science	-0.024	0.023
	(0.021)	(0.021)
socst	-0.039**	0.043**
	(0.020)	(0.020)
femalefemale	0.817**	-0.033
	(0.391)	(0.350)
Constant	1.912*	-4.057***
	(1.127)	(1.223)
Akaike Inf. Crit.	404.070	404.070
Note:		*p**p***p<0.01

NOTE: Use the option `type = "text"` if you want to see the results directly in the RStudio console.

Multinomial logit model: relative risk ratios

```
# Relative risk ratios allow an easier interpretation of the logit coefficients. They are the
exponentiated value of the logit coefficients.

multi1.rrr = exp(coef(multi1))

multi1.rrr
     (Intercept)   science     socst femalefemale
low   6.76855944 0.9767105 0.9618235    2.2627869
high  0.01729593 1.0231865 1.0439413    0.9676623

library(stargazer)
stargazer(multi1, type="html", coef=list(multi1.rrr),  p.auto=FALSE, out="multi1rrr.htm")
```

	Dependent variable:	
	low	high
	(1)	(2)
science	0.977	1.023
	(0.021)	(0.021)
socst	0.962**	1.044**
	(0.020)	(0.020)
femalefemale	2.263**	0.968
	(0.391)	(0.350)
Constant	6.769*	0.017***
	(1.127)	(1.223)
Akaike Inf. Crit.	404.070	404.070
Note:		*p**p***p<0.01

Keeping all other variables constant, if your science score increases one unit, you are 0.97 times more likely to stay in the low ses category as compared to the middle ses category (the risk or odds is 3% lower). The coefficient, however, is not significant.

Keeping all other variables constant, if your science score increases one unit, you are 1.02 times more likely to stay in the high ses category as compared to the middle ses category (the risk or odds is 2% higher). The coefficient, however, is not significant.

NOTE: Use the option `type = "text"` if you want to see the results directly in the RStudio console.

Ordinal logit model: predicted probabilities

```
# At specific values, example science and socst at their means for males and females.
# Use "probs" for predicted probabilities given specific predictors

allmean <-  data.frame(science=rep(mean(mydata$science),2),
                       socst=rep(mean(mydata$socst),2),
                       female = c("male","female"))
allmean
  science  socst female
1   51.85 52.405   male
2   51.85 52.405 female

allmean[, c("pred.prob")] <- predict(multi1, newdata=allmean, type="probs")

allmean
  science  socst female pred.prob.middle pred.prob.low pred.prob.high
1   51.85 52.405   male        0.5555769     0.1441171      0.3003061
2    51.85 52.405 female        0.4739293     0.2781816      0.2478890

# Use "class" for the predicted category
allmean[, c("pred.prob")] <- predict(multi1, newdata=allmean, type="class")

allmean
  science  socst female pred.prob
1   51.85 52.405   male    middle
2   51.85 52.405 female    middle
```

Setup for new predicted probabilities

These are the predicted categories given the new data

Sources

Greene, *Econometric Analysis*, 7th. ed.

Gelman and Hill, *Data Analysis Using Regression and Multilevel/Hierarchical Models*, 2007

UCLA, http://www.ats.ucla.edu/stat/r/dae/

StatsExchange, http://stats.stackexchange.com/

R packages:

-mfx- http://cran.r-project.org/web/packages/mfx/mfx.pdf

-erer- http://cran.r-project.org/web/packages/erer/erer.pdf